**[JDK8新特性：使用Optional避免null导致的NullPointerException](https://blog.csdn.net/aitangyong/article/details/54564100)**

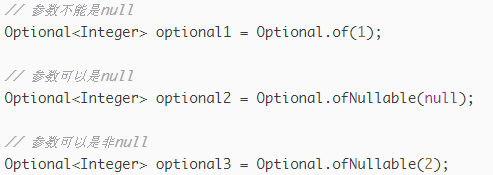
为了解决空指针异常，Google公司著名的Guava项目引入了Optional类，Guava通过使用检查空值的方式来防止代码污染，它鼓励程序员写更干净的代码。受到Google Guava的启发，Optional类已经成为Java 8类库的一部分。

Optional实际是个容器，它不止能保存类型T的值，还能保存Null类型，使用它我们可以不用显式的进行NullPoint判断

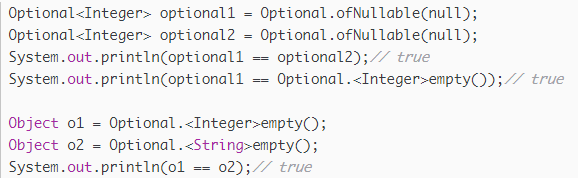
[Optional.of()、Optional.ofNullable()](#of和ofNullable)、[Optional.empty()](#empty)、[isPresent()](#isPresent)、[ifPresent(Consumer consumer)](#ifPresent)、[orElse(value)](#orElse)、[orElseGet(Supplier supplier)](#orElseGet)、[orElseThrow()](#orElseThrow)、[filter(Predicate)](#filter)、[map(Function)](#map)、[flatMap()](#flatMap)

使用静态方法进行Optional（构造方法private）的创建

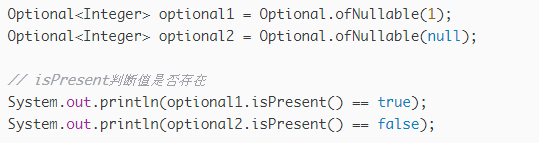
Optional.of()或者Optional.ofNullable()：创建Optional对象，差别在于of不允许**参数**是null，而ofNullable则无限制。



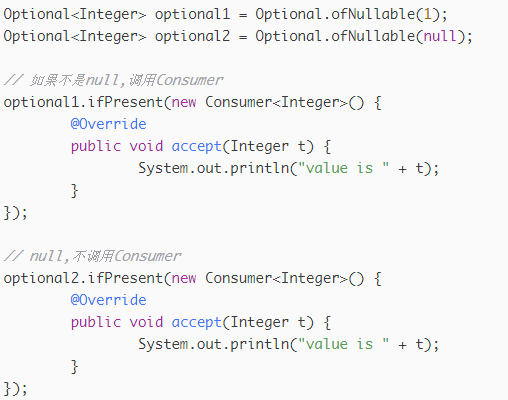
Optional.empty()：所有null包装成的Optional对象：



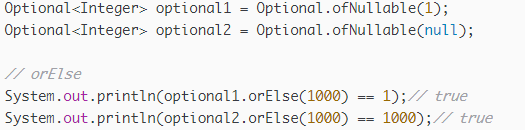
**isPresent()**：判断值是否存在



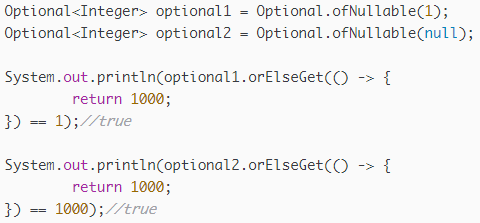
**ifPresent(Consumer consumer)**：如果option对象保存的值不是null，则调用consumer对象，否则不调用



orElse(value)：如果optional对象保存的值不是null，则返回原来的值，否则返回value



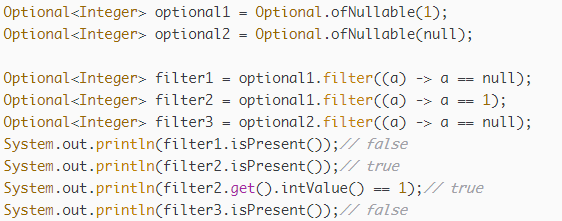
orElseGet(Supplier supplier)：功能与orElse一样，只不过orElseGet参数是一个对象



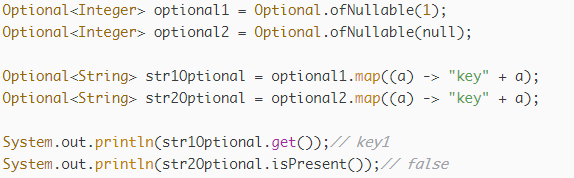
orElseThrow()：值不存在则抛出异常，存在则什么不做，有点类似Guava的Precoditions



filter(Predicate)：判断Optional对象中保存的值是否满足Predicate，并返回新的Optional



map(Function)：对Optional中保存的值进行函数运算，并返回新的Optional(可以是任何类型)



flatMap方法与map方法类似，区别在于mapping函数的返回值不同。map方法的mapping函数返回值可以是任何类型T，而flatMap方法的mapping函数必须是Optional

